Terms used
'shared memory ' and **'local memory'** and **instruction** and **constraint**

Found **12,513** of **132,857**

Sort results by: relevance

Display results: expanded form

Save results to a Binder

Search Tips

Open results in a new window

Try an Advanced Search

Try this search in The ACM Guide

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next

Best 200 shown

Relevance scale

1 Memory access buffering in multiprocessors

M. Dubois, C. Scheurich, F. Briggs

June 1986 **ACM SIGARCH Computer Architecture News , Proceedings of the 13th annual international symposium on Computer architecture**, Volume 14 Issue 2

Full text available: pdf(943.66 KB) Additional Information: full citation, abstract, references, citings, index terms

In highly-pipelined machines, instructions and data are prefetched and buffered in both the processor and the cache. This is done to reduce the average memory access latency and to take advantage of memory interleaving. Lock-up free caches are designed to avoid processor blocking on a cache miss. Write buffers are often included in a pipelined machine to avoid processor waiting on writes. In a shared memory multiprocessor, there are more advantages in buffering memory requests, since each m ...

2 User-controllable coherence for high performance shared memory multiprocessors

Collin McCurdy, Charles Fischer

June 2003 **ACM SIGPLAN Notices , Proceedings of the ninth ACM SIGPLAN symposium on Principles and practice of parallel programming**, Volume 38 Issue 10

Full text available: pdf(210.24 KB) Additional Information: full citation, abstract, references, index terms

In programming high performance applications, shared address-space platforms are preferable for fine-grained computation, while distributed address-space platforms are more suitable for coarse-grained computation. However, currently only distributed address-space systems scale beyond the low hundreds of processors. In this paper we introduce a hybrid architecture that allows users to trade off local memory usage for coherence communication, making possible larger-scale shared memory architecture ...

Keywords: distributed memory architectures, irregular computation, parallel computation, shared memory architectures

3 Is SC + ILP = RC?

Chris Gniady, Babak Falsafi, T. N. Vijaykumar

May 1999 **ACM SIGARCH Computer Architecture News , Proceedings of the 26th annual international symposium on Computer architecture**, Volume 27 Issue 2

Full text available: pdf(94.82 KB) Additional Information: full citation, abstract, references, citings, index terms

Publisher Site

Sequential consistency (SC) is the simplest programming interface for shared-memory systems but imposes program order among all memory operations, possibly precluding high performance implementations. Release consistency (RC), however, enables the highest

performance implementations but puts the burden on the programmer to specify which memory operations need to be atomic and in program order. This paper shows, for the first time, that SC implementations can perform as well as RC implementations ...

4 Empirical evaluation of the CRAY-T3D: a compiler perspective

Remzi H. Arpaci, David E. Culler, Arvind Krishnamurthy, Steve G. Steinberg, Katherine Yelick

May 1995 **ACM SIGARCH Computer Architecture News , Proceedings of the 22nd annual international symposium on Computer architecture**, Volume 23 Issue 2

Full text available: pdf(1.48 MB) Additional Information: full citation, abstract, references, citings, index terms

Most recent MPP systems employ a fast microprocessor surrounded by a shell of communication and synchronization logic. The CRAY-T3D provides an elaborate shell to support global-memory access, prefetch, atomic operations, barriers, and block transfers. We provide a detailed empirical performance characterization of these primitives using micro-benchmarks and evaluate their utility in compiling for a parallel language. We have found that the raw performance of the machine is quite impressive and ...

5 Efficient synchronization of multiprocessors with shared memory

Clyde P. Kruskal, Larry Rudolph, Marc Snir

October 1988 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 10 Issue 4

Full text available: pdf(1.78 MB) Additional Information: full citation, abstract, references, citings, index terms, review

A new formalism is given for read-modify-write (RMW) synchronization operations. This formalism is used to extend the memory reference combining mechanism introduced in the NYU Ultracomputer, to arbitrary RMW operations. A formal correctness proof of this combining mechanism is given. General requirements for the practicality of combining are discussed. Combining is shown to be practical for many useful memory access operations. This includes memory updates of the form mem_ ...

6 The interaction of software prefetching with ILP processors in shared-memory systems

Parthasarathy Ranganathan, Vijay S. Pai, Hazim Abdel-Shafi, Sarita V. Adve

May 1997 **ACM SIGARCH Computer Architecture News , Proceedings of the 24th annual international symposium on Computer architecture**, Volume 25 Issue 2

Full text available: pdf(2.44 MB) Additional Information: full citation, abstract, references, citings, index terms

Current microprocessors aggressively exploit instruction-level parallelism (ILP) through techniques such as multiple issue, dynamic scheduling, and non-blocking reads. Recent work has shown that memory latency remains a significant performance bottleneck for shared-memory multiprocessor systems built of such processors.This paper provides the first study of the effectiveness of software-controlled non-binding prefetching in shared memory multiprocessors built of state-of-the-art ILP-based proces ...

7 Java consistency: nonoperational characterizations for Java memory behavior

Alex Gontmakher, Assaf Schuster

November 2000 **ACM Transactions on Computer Systems (TOCS)**, Volume 18 Issue 4

Full text available: pdf(305.72 KB) Additional Information: full citation, abstract, references, citings, index terms

The Java Language Specification (JLS) [Gosling et al. 1996] provides an operational definition for the consistency of shared variables. The definition remains unchanged in the JLS 2nd edition, currently under peer review, which relies on a specific abstract machine as its underlying model, is very complicated. Several subsequent works have tried to simplify and formalize it. However, these revised definitions are also operational, and thus have failed to highlight the intuition behind the o ...

Keywords: Java memory models, multithreading, nonoperational specification

8 Tempest and typhoon: user-level shared memory
S. K. Reinhardt, J. R. Larus, D. A. Wood
April 1994 **ACM SIGARCH Computer Architecture News , Proceedings of the 21ST annual international symposium on Computer architecture**, Volume 22 Issue 2

Full text available: pdf(1.44 MB) Additional Information: full citation, abstract, references, citings, index terms

Future parallel computers must efficiently execute not only hand-coded applications but also programs written in high-level, parallel programming languages. Today's machines limit these programs to a single communication paradigm, either message-passing or shared-memory, which results in uneven performance. This paper addresses this problem by defining an interface, *Tempest*, that exposes low-level communication and memory-system mechanisms so programmers and compilers can customize polici ...

9 Integrating message-passing and shared-memory: early experience
David Kranz, Kirk Johnson, Anant Agarwal, John Kubiatowicz, Beng-Hong Lim
July 1993 **ACM SIGPLAN Notices , Proceedings of the fourth ACM SIGPLAN symposium on Principles and practice of parallel programming**, Volume 28 Issue 7

Full text available: pdf(1.12 MB) Additional Information: full citation, abstract, references, citings, index terms

This paper discusses some of the issues involved in implementing a shared-address space programming model on large-scale, distributed-memory multiprocessors. While such a programming model can be implemented on both shared-memory and message-passing architectures, we argue that the transparent, coherent caching of global data provided by many shared-memory architectures is of crucial importance. Because message-passing mechanisms ar much more efficient than shared-memory loads and stores fo ...

10 Parallel logic programming systems
Jacques Chassin de Kergommeaux, Philippe Codognet
September 1994 **ACM Computing Surveys (CSUR)**, Volume 26 Issue 3

Full text available: pdf(3.51 MB) Additional Information: full citation, abstract, references, citings, index terms

Parallelizing logic programming has attracted much interest in the research community, because of the intrinsic OR- and AND-parallelisms of logic programs. One research stream aims at transparent exploitation of parallelism in existing logic programming languages such as Prolog, while the family of concurrent logic languages develops language constructs allowing programmers to express the concurrency—that is, the communication and synchronization between parallel processes—withi ...

Keywords: AND-parallelism, OR-parallelism, Prolog, Warren Abstract Machine, binding arrays, concurrent constraint programming, constraints, guard, hash windows, load balancing, massive parallelism, memory management, multisequential implementation techniques, nondeterminism, scheduling parallel tasks, static analysis

11 Data and memory optimization techniques for embedded systems
P. R. Panda, F. Catthoor, N. D. Dutt, K. Danckaert, E. Brockmeyer, C. Kulkarni, A. Vandercappelle, P. G. Kjeldsberg
April 2001 **ACM Transactions on Design Automation of Electronic Systems (TODAES)**, Volume 6 Issue 2

Full text available: pdf(339.91 KB) Additional Information: full citation, abstract, references, citings, index terms

We present a survey of the state-of-the-art techniques used in performing data and memory-related optimizations in embedded systems. The optimizations are targeted directly or indirectly at the memory subsystem, and impact one or more out of three important cost metrics: area, performance, and power dissipation of the resulting implementation. We first examine architecture-independent optimizations in the form of code transoformations. We

next cover a broad spectrum of optimizati ...

Keywords: DRAM, SRAM, address generation, allocation, architecture exploration, code transformation, data cache, data optimization, high-level synthesis, memory architecture customization, memory power dissipation, register file, size estimation, survey

12 Communication synthesis for distributed embedded systems

Ti-Yen Yen, Wayne Wolf

December 1995 **Proceedings of the 1995 IEEE/ACM international conference on Computer-aided design**

Full text available: pdf(205.25 KB) Publisher Site Additional Information: full citation, abstract, references, citings, index terms

Abstract: Communication synthesis is an essential step in hardware-software co-synthesis: many embedded systems use custom communication topologies and the communication links are often a significant part of the system cost. This paper describes new techniques for the analysis and synthesis of the communication requirements of embedded systems during co-synthesis. Our analysis algorithm derives delay bounds on communication in the system given an allocation of messages to links. This analysis al ...

Keywords: CAD, analysis algorithm, co-synthesis, communication links, delay bounds, distributed embedded systems, distributed processing, embedded systems, hardware-software co-synthesis, interprocess communication, real-time systems, synthesis algorithm, system buses

13 The NYU Ultracomputer---designing a MIMD, shared-memory parallel machine (Extended Abstract)

Allan Gottlieb, Ralph Grishman, Clyde P. Kruskal, Kevin P. McAuliffe, Larry Rudolph, Marc Snir

April 1982 **Proceedings of the 9th annual symposium on Computer Architecture**

Full text available: pdf(1.36 MB) Additional Information: full citation, abstract, references, citings, index terms

We present the design for the NYU Ultracomputer, a shared-memory MIMD parallel machine composed of thousands of autonomous processing elements. This machine uses an enhanced message switching network with the geometry of an Omega-network to approximate the ideal behavior of Schwartz's paracomputer model of computation and to implement efficiently the important fetch-and-add synchronization primitive. We outline the hardware that would be required to build a 4096 processor system using 1990' ...

14 Mark IIIfp hypercube concurrent processor architecture

J. Tuazon, J. Peterson, M. Pniel

January 1988 **Proceedings of the third conference on Hypercube concurrent computers and applications: Architecture, software, computer systems, and general issues - Volume 1**

Full text available: pdf(1.02 MB) Additional Information: full citation, abstract, references, citings, index terms

The Mark IIIfp Hypercube is a new generation of hypercube concurrent processor system developed at JPL/Caltech, with peak performance of 5 Mips, 14 Mflops per node, and a peak communication rate of 6 Mbytes per second. Each node utilizes two Motorola MC68020 microprocessors, an MC68882 scalar floating- point coprocessor, and a Weitek 8000 floating-point chip set. One of the MC68020 processors serves as the application and computational processor, the other is dedicated to communication. The ...

15 Efficient synchronization of multiprocessors with shared memory

Clyde P Kruskal, Larry Rudolph, Marc Snir

November 1986 **Proceedings of the fifth annual ACM symposium on Principles of distributed computing**

h c g e cf c

Full text available: pdf(897.28 KB) Additional Information: full citation, references, citings, index terms

16 Tempest and typhoon: user-level shared memory
Steven K. Reinhardt, James R. Larus, David A. Wood
August 1998 **25 years of the international symposia on Computer architecture (selected papers)**
Full text available: pdf(1.57 MB) Additional Information: full citation, references, index terms

17 The gould NP1 system interconnecting
D. J. Vianney, J. H. Thomas, V. Rabaza
June 1988 **Proceedings of the 2nd international conference on Supercomputing**
Full text available: pdf(1.28 MB) Additional Information: full citation, abstract, references, index terms

The Gould NP1 is a multicomputer multiprocessing system designed for high performance and parallel processing required in diverse scientific and engineering applications. The NP1's basic building block is a dual-processor single bus system which can be expanded up to eight processors over four system buses. This paper discusses the overall design and implementation of tne NP1 system interconnection in particularly the inter-system bus link which interconnects four system buses to ...

18 Shasta: a low overhead, software-only approach for supporting fine-grain shared memory
Daniel J. Scales, Kourosh Gharachorloo, Chandramohan A. Thekkath
October 1996 **Proceedings of the seventh international conference on Architectural support for programming languages and operating systems**, Volume 30 , 31 Issue 5 , 9
Full text available: pdf(1.49 MB) Additional Information: full citation, abstract, references, citings, index terms

This paper describes Shasta, a system that supports a shared address space in software on clusters of computers with physically distributed memory. A unique aspect of Shasta compared to most other software distributed shared memory systems is that shared data can be kept coherent at a fine granularity. In addition, the system allows the coherence granularity to vary across different shared data structures in a single application. Shasta implements the shared address space by transparently rewrit ...

19 Experience Using Multiprocessor Systems—A Status Report
Anita K. Jones, Peter Schwarz
June 1980 **ACM Computing Surveys (CSUR)**, Volume 12 Issue 2
Full text available: pdf(4.48 MB) Additional Information: full citation, references, citings, index terms

20 Cache coherence in large-scale shared-memory multiprocessors: issues and comparisons
David J. Lilja
September 1993 **ACM Computing Surveys (CSUR)**, Volume 25 Issue 3
Full text available: pdf(3.12 MB) Additional Information: full citation, references, citings, index terms

h c g e cf c